CptS 515 Advanced Algorithms HOMEWORK-3

Sharath Kumar Karnati

October 2023

QUESTION 1:

Let D be a device that keeps sending out messages. Each message contains two parts A and B where the intruder can not observe A but he can observe B. Suppose that each of A and B takes 10 bits so, each message is exactly 20 bits. Before the developers sell the device to the public (including the intruder), they run some experiments trying to make sure that there isn't any information leakage that is more than 1 bit from A to B in a message.The experiments are done by run the device for a long time and obtain a large and finite set C of messages. Please design a program that can estimate the average number of bits actually leaked from A to B in a message drawn from C.

SOLUTION:

For this problem, we will solve by connecting bipartite matching to information flow from A to B. In Bipartite graph, We define with two sets of nodes V1 and V2, which represents A and B. We will take one node for each bit in message. Each edge defines a binary relationship between the two nodes,V1 and V2, indicating whether the node in V2's value leaks information about the node in V1's value.We can design one bipartite matching problem for each message in C and compute the average of the outcomes across all messages to find the average number of bits that leak from A to B.

Algorithm for this problem:

Aim: To find average number of bits that are leaked from A to B.
Input: A set of finite messages C, categorized into two components A and B.
Output: Average number of bits leaked from A to B.
Step 1: leakedbits = 0
Step 2: For each message x in C then do
Step 3: Now, construct a bipartite matching problem M to model the message x.
Step 4: Result = Network flow based solution to M (Bipartite Matching Problem).
Step 5: leakedbits = leakedbits + result. or leakedbits+= result
Step 6 : End for loop
Step 7: Return leakedbits.

QUESTION 2: Consider a C-function that has integer variables as arguments and integer as return type: int myFunction(int x1, int x2, ..., int x7) In the function with arguments x1,x2,...,x7 which are integer variables, there are only 10 lines of code, where each line is in the form of an assignment variable := Exp to an integer variable where Exp is a linear combi-nation of integer variables (e.g., y:=2x1+3x2-5) or an if-then-else statement where the condition is a comparison between two linear constraints on integer variables and the assignments in the if-then-else statement are in the form of variable := Exp shown above (e.g., if (y¿12x1-z) then x2:=3x7-15 else x5:=18x4-6x7+6. The first line of the function declares three integer variable x,y,z, while the last line is to return the value x back. Please design a program that can verify whether there are values for x1,x2,...,x7 passed to the function that can make the function return a negative integer.

SOLUTION:

For this given problem, we can solve it by reduction into Integer Linear Programming (ILP). In ILP formulation, we will generally have Objective Function and Constraints as the main components.So, now we will set our objective function as "minimumx" subject to the following constraints.

- Firstly, we will set the expression as a constraint like an assignment for each odd line.For Example: given, $y = 2x_1 + 3x_2 - 5$, will be converted as an equivalent constraint $y - 2x_1 - 3x_2 = -5$, Due to the fact that all such expressions are linear combinations of integer variables , so, the related constraints will also be linear.
- Now,each if-then-else statement in a line of code can be converted into 2 equivalent logical implication statements .These implication assertions can be converted into linear integer constraints using the Big-M-formulation.Example : If we consider, If-else-then statements; such as ; if $(y > 12x_1 - z)$ then $x_2 := 3x_7 - 15$, now we will create two constraints: as :a.) condition: Ex: $y - 12x_1 + z \leq 0$ and b.)The assignment : Ex: $x_2 - 3x_7 + 15 \leq 0$ and we will formulate for all lines of code in the function.
- Now we will set objective function(minimumx) to minimize the value of x. This will be our goal to find values which will make a negative integer.
- Now we will use any traditional ILP solver to solve this problem

For this ILP, we can use any conventional solver to find the solution. We can check that there are some conceivable combinations of $x1, x2, \ldots.., x7$ that can result in the function returning negative if the solver's optimal value is less than zero.

QUESTION 3: Symbolic representation is way to code a finite object. BDD is a way to code a finite set. However, when a power set (a set of finite sets) is given,BDD is not usually efficient. Sometimes, it is a good idea to code an object as a number since a number itself is a string (e.g., 123 is the string "123").We now consider a special case. Let K = 1, ..., k for some k, and consider P be a set of disjoint subsets of K. That is, P = K1, · · · , Km for some m and each Ki K and Ki Kj = whenever i 6 = j. I want to design an algorithm, for a given K, to code (or transform or represent) each such P into a number CP such that the code C is optimal; i.e., (1). C is 1-1, (2). CP 1, ..., BK , where BK is the number of all such P 's for the given K.

SOLUTION:

We will use Boolean formula with representation of graphs to solve this problem.

Now, The K will be a power set of k values which will be $1, 2, \ldots\ldots, k$ that will contain 2K elements that can be represented with a boolean variable of size K i.e. $(x_1, x_2, \ldots., x_k\ )\ \forall\ x_i \in (0, 1)$. Now every assignment of $(x_1, \ldots., x_k)$ will act as a subset of the power set K. For example: $(1, 2, 3)$ subset will be represented as $x_1 = 1, x_2 = 2, x_3 = 3$ and $\forall\ x_i = 0$ and $i > 3$.

Now we will assume that the graph as G which has its vertex set containing all subsets of K. This can be represented by boolean variables of k $(x_1, \ldots x_k)$.

Now, we will connect each pair of disjoint subsets with an edge. The edge set of G will be represented with a Boolean Formula $R((x_1, ..., x_k), (y_1, \ldots, y_k)) = (\neg(x_1 \wedge y_1) \wedge (\neg(y_2 \wedge x_2)) \ldots\ldots (\neg(x_k \wedge y_k)))$, here set of x and y acts as two arbitrary nodes of G.

Now, with the help of G, we will iteratively generate a code for every possible set P as following:

- 1. Firstly , we will construct an empty list E.
- 2. Now, the index of each item in E will represent a code for each possible$P = (K_1, \ldots, K_m)$, where m represents number of subsets of K that are in P.
- 3. Then, For m=1 , we will append all subsets of K to E. Here , these subsets will be represented by a vertex set of G. This will provide us an integer code for each P that has one element.
- 4. Now, For m=2, we will append all pairs of subsets K_1, K_2. Then these will be computed by the edge set G, that is, nodes for which $R((x_1, , , , x_k), (y_1, \ldots, y_k))$ is satisfied.
- 5. Now, For $m \geq 3$, we will append all the nodes that will occur on a walk length m-1 from the source $(x_1, \ldots., x_k)$to the goal $(y_1, \ldots., y_k)$ nodes which will satisfy this boolean expression: $R' = R \wedge (RoR) \wedge \ldots\ldots\ldots \wedge (RoRoR\ldots\ldots(m-1)times)$.
- 6. From the above expression, we can know that every node on this path should be a pairwise disjoint. As m is finite, then this process will also be finite and cannot last forever.

Hence, the above mentioned process populates a list with each possible P and the corresponding index gives an optimal code Cp.

QUESTION 4: (easy) Let G be a directed graph of 2048 nodes. When we use a Boolean formula to represent the G, how many Boolean variables are needed in the formula?

SOLUTION:
Given , G is a directed graph with 2048 nodes.
$2^x = 2048$
$2^x = 2^{11}$
x=11
So, then we will have 11 boolean variables for 2048 node but we will require 22 boolean variables for each edge when G is represented by a Boolean formula.
Hence, 22 boolean variables are needed in this formula.

QUESTION 5: (easy) Data types are an abstraction of data and data structures are a way to store the types into memory. In particular, we never store physical objects in memory; in case when we really want to do it, we first represent the physical objects in an abstract representation and store the representation in memory. Here is a problem. There are 40 students in a classroom. I want to design a closet so that any one of the students can be hidden inside.Imagine that the closet is a chunk of computer memory. Then, how big (in bits) closet do you need?
Solution:
Given , Number of students= 40,
As we will have 40 different students that are needed to be stored in closet, so there will be 40 unique objects.
Then , Number of bits =$\log_2 40$= 5.32192. . .
So, we will require approximately 6 bits to store 40 students.
